KAPITEL 17
Druckerkonfiguration: KCONFIG.PRG

Verschiedene auf dem Markt befindliche Drucker verarbeiten die ihnen gesendeten Zeichen unterschiedlich. Mit dem Programm K-Spread 4 wird eine Standard-Druckeranpassung geliefert, die mit Epson FX-80 kompatiblen Druckern zusammenarbeitet. Wenn Ihr Drucker nicht zu dieser Kategorie gehört und nicht korrekt arbeitet, können Sie eine eigene Datei zur Druckereinstellung erzeugen. Ihr Druckerhandbuch sollten Sie dabei stets griffbereit haben.

Lesen Sie auch die Beschreibungen im DRUCK-Menü, denn K-Spread 4 erlaubt viele Einstellungen, ohne das Programm beenden und KCONFIG.PRG aufrufen zu müssen.

Die Einstellungen, die Sie durch KCONFIG.PRG vornehmen, werden erst wirksam, wenn Sie die Datei zur Druckeranpassung in K-Spread geladen haben.

Nach dem Start des Programms KCONFIG.PRG vom GEM-Desktop aus können Sie anwählen, ob Sie eine Druckereinstellungsdatei neu erzeugen, eine bestehende laden, eine im Speicher befindliche ändern oder abspeichern wollen.

Wenn Sie eine neue anlegen, erscheinen die folgenden Optionen:

Druckername

Geben Sie den Namen des Druckers an, für den Sie die neue Druckeranpassung erstellen. Diese Eingabe ist noch nicht der spätere Dateiname.

Port

Sie werden danach gefragt, ob der Drucker über den parallelen oder seriellen Port angeschlossen ist. Bei seriellem Anschluß können Sie weitere Optionen einstellen.

Papiereinstellung

Die einzelnen Wahlmöglichkeiten sind selbsterklärend oder an anderer Stelle des Handbuchs beschrieben.

Steuerzeichen

Sie können den aufgeführten Kommandos, welche K-Spread an den Drucker sendet, die entsprechenden Steuerzeichen zuweisen. Sehen Sie dazu in ihrem Druckerhandbuch nach. Die Steuerzeichen können Sie in folgenden Formen verwenden:

- Eine Zahl zwischen 0 und 255 repräsentiert den dezimalen ASCII-Code
- Im hexadezimalen Zahlensystem stellt Zahl mit einem "$" direkt davor den Code dar ($0 .. $FF)
- Ein in Anführungszeichen eingeschlossenes Zeichen, von " " (gleichbedeutend mit ASCII-Zeichen 32 (=Space)) bis "~" (äquivalent mit ASCII 126)
- Einer der Standard-Steuerzeichen, z.B. LF, STX, ETX, etc.

Sie können bis zu 10 Codes verwenden, die jeweils durch ein "Space" voneinander getrennt sind.

Konvertiertabelle

Ein vom Anwendungsprogramm gesendetes Zeichen wird anhand der Konvertiertabelle umgewandelt. Ein einzelner Code kann durch eine Sequenz von bis zu 10 Kommandos ersetzt werden. Diese Zeichen werden dann am Drucker ausgegeben.

Klicken Sie mit der Maus einfach in der Tabelle das Zeichen an, für das ein anderes Zeichen oder eine Kommando-Sequenz gedruckt werden soll. In einer Dialogbox können Sie dies auf die vier oben beschriebenen Arten eingeben oder ändern.

KAPITEL 18
GDOS

K-Spread unterstützt die Verwendung von GDOS (Graphical Device Operating System). Mit GDOS können Sie gute Schriften und Grafiken mit verschiedenen Druckern erzeugen, mit Bildschirmen, Plottern, Kameras u.a.. Eine kurze Beschreibung folgt.

Im AUTO-Ordner Ihrer Festplatte/Boot-Diskette muß sich das Programm GDOS.PRG befinden, das dann nach jedem Reset des Rechners geladen wird. GDOS.PRG benötigt eine Datei namens ASSIGN.SYS, die sich im Hauptverzeichnis der Festplatte/Boot-Diskette befinden muß (C:\ oder A:\). ASSIGN.SYS enthält Informationen darüber, wo GDOS seine Fonts und Treiber findet.

Wenn Sie eine Festplatte besitzen, geht die ganze Installation leicht. Wenn Sie jedoch nur ein doppelseitiges Diskettenlaufwerk besitzen, sollten Sie versuchen, alle Zeichensätze für einen einzigen Drucker auf einer Diskette unterzubringen.

Wenn Sie lediglich ein einseitiges Diskettenlaufwerk besitzen, sollten Sie sich auf einen Font beschränken, oder auf wenige 'kleine' Fonts. Denn der Umfang der Zeichensatzdateien stallt dann ein Problem dar, da sie auf mehrere Disketten verteilt sein können. Unglücklicherweise sind Sie angewiesen, ALLE Fonts auf einer einzigen Diskette unterzubringen.

Sie können die verschiedenen Zeichensätze aufgrund ihrer Dateinamen erkennen. Die ersten vier Buchstaben stellen den Namen dar, die letzten zwei Ziffern die Größe. Alle Zeichensatzdateien haben die Endung .FNT. K-Spread 4 erlaubt es Ihnen, Rechenblätter mit großen Fonts zu zeichnen. Wenn Sie jedoch wenig Speicher besitzen, können Sie die großen Zeichensätze von der Diskette und aus ASSIGN.SYS entfernen.

Jeder Zeichensatz, den Sie benutzen möchten, muß in der Datei ASSIGN.SYS verzeichnet sein. Und jeder Eintrag in ASSIGN.SYS sollte sich auch wirklich auf Festplatte/Diskette befinden.

In ASSIGN.SYS sind zudem alle Treiber aufgeführt, die das Ansteuern des entsprechenden Ausgabegeräts ermöglichen. Diese Gerätenamen beginnen mit einer Nummer und enden mit .SYS.

Eine typische ASSIGN.SYS-Datei könnte wie auf der folgenden Seite aussehen.

PATH = A:\GDOS.SYS; Pfad, über den GDOS seine Fonts und Treiber findet

1p SCREEN.SYS; Bildschirm-Zeichensätze des Default-Screens

ATSS10.FNT
ATSS12.FNT
ATSS18.FNT
ATSS24.FNT
ATTP10.FNT
ATTR10.FNT
ATTR12.FNT
ATTR18.FNT
ATTR24.FNT

2p SCREEN.SYS; Bildschirm-Zeichensätze für die niedrige (LOW) Auflösung

ATSS10.FNT
ATSS12.FNT
ATSS18.FNT
ATSS24.FNT
ATTP10.FNT
ATTR10.FNT
ATTR12.FNT
ATTR18.FNT
ATTR24.FNT

3p SCREEN.SYS; Bildschirm-Zeichensätze für die mittlere (MEDIUM) Auflösung

ATSS10CG.FNT
ATSS12CG.FNT
ATSS18CG.FNT
ATSS24CG.FNT
ATTP10CG.FNT
ATTR10CG.FNT
ATTR12CG.FNT
ATTR18CG.FNT
ATTR24CG.FNT

4p SCREEN.SYS; Bildschirm-Zeichensätze für die hohe (HIGH) Auflösung

ATSS10.FNT
ATSS12.FNT
ATSS18.FNT

ATSS24.FNT
ATTP10.FNT
ATTR10.FNT
ATTR12.FNT
ATTR18.FNT
ATTR24.FNT

21 FX80.SYS; Druckerzeichensätze

ATSS10EP.FNT
ATSS12EP.FNT
ATSS18EP.FNT
ATSS24EP.FNT
ATTP10EP.FNT
ATTR10EP.FNT
ATTR12EP.FNT
ATTR18EP.FNT
ATTR24EP.FNT

31 META.SYS; Zeichensätze für das Metafile

ATSS10MF.FNT
ATSS12MF.FNT
ATSS18MF.FNT
ATSS24MF.FNT
ATTP10MF.FNT
ATTR10MF.FNT
ATTR12MF.FNT
ATTR18MF.FNT
ATTR24MF.FNT

Wie Sie sicherlich bemerkt haben, befindet sich am Anfang jedes Blocks eine Nummer, z.B. 21 FX80.SYS. Sie werden dazu benutzt, die Ausgabegeräte anzusprechen:

0 ..10: Bildschirme. Kein Treiber nötig.
11..20: Plotter. Kein Treiber mitgeliefert.
21..30: Drucker. 2 Treiber mitgeliefert (NB15/NEC P6 und FX80)
31..40: Metafiles. Treiber mitgeliefert.
41..50: Cameras. Kein Treiber mitgeliefert.

Selbstverständlich können Sie auch mehrere ähnliche Ausgabegeräte aufführen, z.B.:

...
21 FX80.SYS ;Erster Drucker, kompatibel zu Epson FX80
... Fonts für FX80 ...
22 NB15.SYS ;Zweiter Drucker, kompatibel zu NEC P6
... Fonts für NB15 ...

In K-Spread können Sie vor einem GDOS-Ausdruck die Gerätenummer eingeben. Hier also beispielsweise 21 für Ihren FX80-kompatiblen (Probeausdruck), und später 22 für Ihren Laserdrucker.
Metafiles sind Dateien, die Zeichenanweisungen enthalten. Andere Programme können solche Dateien lesen und die Grafik weiterverarbeiten. Ein ideale Anwendung ist das Einbinden eines Rechenblatts oder Graphen in ein Desktop-Publishing-Programm (DTP). "Calamus", "Timeworks Publisher" oder "Easydraw" können Metafiles einlesen. Das mitgelieferte Programm OUTPUT.PRG dient dazu, Metafiles auf dem Bildschirm oder Drucker auszugeben.
Eine der mitgelieferten ASSIGN-Dateien sollten Sie in ASSIGN.SYS umbenennen. Eine solche Datei können Sie mit einem Editor oder einer Textverarbeitung nach Ihren Wünschen verändern.

KAPITEL 19
Macros

Es wird öfters vorkommen, da Sie bei unterschiedlichen Arbeiten die gleiche Abfolge von Arbeitsschritten durchführen müssen. Als Beispiel sei das Berechnen und Drucken von Investment-Papieren genannt, die entsprechend ihrem Kapitalwert einen bestimmten Zinssatz und eine gewisse Laufzeit aufweisen. Wollen Sie nun verschiedene Wertpapiere mit verschieden hohen Geldbeträgen durchkalkulieren und ausdrucken wollen, so macht dies normalerweise recht viel Arbeit. Für jedes Wertpapier haben Sie genau dieselben Arbeitsgänge.
Hier erscheint ein Silberstreif am Horizont: Macros. In unserem Fall können Sie in einem Macro die nötigen Arbeitsschritte für Berechnung und Druck der Investment-Papiere angeben. Sie rufen einfach dieses Macro auf, das nun die ganze Arbeit erledigt, während Sie sich (anspruchsvolleren) Dingen widmen können.
Ein Macro-Kommando ist im Grunde nichts anderes als eine spezielle Funktion. Die Ausführung eines solchen Kommandos bewirkt jedoch nicht beispielsweise das Berechnen der Quadratwurzel, sondern löst zum großen Teil Aktionen aus, die Sie auch über die Menüzeile erreichen können. In der Schreibweise stimmen Macro-Befehle und die eingebauten Funktionen im wesentlichen überein.
Im folgenden werden Sie jeweils anhand eines Beispiels zwei Arten kennenlernen, wie Sie Macros erzeugen können.
Danach folgt ein kurzer Abschnitt, in dem der Aufbau der Macro-Befehle beschrieben wird. Sie erfahren auch, welche Bedeutung die Einträge im MACRO-Menü der Hauptmenüzeile haben. Viel Raum nimmt im Anschluß daran der Nachschlageteil ein, in welchem ausführlich jeder Macro-Befehl erklärt wird.
Es wird davon ausgegangen, daß Sie zuerst die Beispiele bearbeitet haben, bevor Sie richtig experimentieren.

Das Erzeugen eines Macros

A. Benutzung des Aktionen-Recorders

Macros können am einfachsten mittels eines "Recorders" erzeugt werden, der den gesamten Prozeß automatisiert - indem er die Aktionen des Benutzers 'aufzeichnet'. Das folgende Beispiel wird das verdeutlichen:

1) Legen Sie ein neues Blatt an (ARBEITEN-Menü, "Blatt anlegen"). Klicken Sie in der Dialogbox, in welcher Sie den Namen eingeben, bitte "Macro" an. Sie haben damit ein sogenanntes "Macro-Blatt" erzeugt, das die Endung .SPM hat, und sich wie ein normales Rechenblatt bedienen läßt. In dieses Blatt können Sie mehrere Macros eingeben bzw. aufzeichnen.
2) Klicken Sie "Macro anlegen" an (MACRO-Menü). Geben Sie einen (möglichst aussagefähigen) Namen für Ihr Macro ein. Bestätigen Sie mit "OK".
3) Die bekannte Eingabezeile zum Bestimmen eines Bereichs erscheint, in die Sie das Startfeld des Macros eingeben, in unserem Fall A0:A0. Mit OK/Return bestätigen.

   Achten Sie bei der bereits bekannten Abfrage darauf, daß Sie nur das erste Feld und nicht alle Felder des Macros angeben.
4) Setzen Sie den Edit-Rahmen auf das Feld, in dem das Macro beginnen soll. Klicken Sie auf "Macroanfang setzen.." im MACRO-Menü.
5) Aktivieren Sie ein Rechenblatt-Fenster, setzen sie den Edit-Rahmen auf ein Startfeld und selektieren Sie "starte Aufnahme" im MACRO-Menü. Führen Sie nun die Aktionen aus, die später vom Macro ausgeführt werden sollen. So können Sie beispielsweise in Feld D5 den Text "Hallo Welt" eingeben, zusätzlich vielleicht noch eine Zahl in ein anderes Feld.

   Wenn Sie alle gewünschten Arbeitsgänge ausgeführt haben, selektieren Sie "beende Aufnahme" im MACRO-Menü.
6) Sollte das oberste Fenster kein Rechenbatt-Fenster (Endung .SPD) sein, so aktivieren Sie eines, indem Sie an eine beliebige Stelle innerhalb des gewünschten Rechenblattes klicken.
7) Selektieren Sie "Macro ausführen.." im MACRO-Menü und wählen Sie aus der Liste das benötigte Macro aus.

Im Gegensatz zur direkten Eingabe der Macro-Befehle beginnt hier die Ausgabe da, wo der Edit-Rahmen steht. Die Positionen von Text und Zahl werden relativ zueinander angegeben.

B. Direkte Eingabe der Macro-Befehle

1) Erzeugen Sie wie oben ein neues Macro-Blatt, oder bringen Sie durch Anklicken ein Macro-Fenster ganz nach oben.
2) Geben Sie nun eine Liste mit Macro-Anweisungen in aufeinanderfolgende Felder einer Spalte ein. Das letzte Feld erhält den Befehl "=RETURN()". Das Macro könnte z.B. so aussehen:

	A	B	C
0	=SELECT(D5)		
1	=FORMULA("Hallo Welt")		
2	=RETURN()		

3) Geben Sie einen (möglichst aussagefähigen) Namen für Ihr Macro ein (MACRO-Menü, "Macro anlegen"). Bestätigen Sie mit "OK".
4) Die bekannte Eingabezeile zum Bestimmen eines Bereichs erscheint, in die Sie das Startfeld des Macros eingeben, in unserem Fall A0:A0. *Achten Sie bei der bereits bekannten Abfrage darauf, daß Sie nur das erste Feld und nicht alle Felder des Macros angeben.*
5) Sollte das oberste Fenster kein Rechenbatt-Fenster (Endung .SPD) sein, so aktivieren Sie eines, indem Sie an eine beliebige Stelle innerhalb des gewünschten Rechenblattes klicken.
6) Selektieren Sie "Macro ausführen.." im MACRO-Menü und wählen Sie aus der Liste das benötigte Macro aus.
7) Als Endergebnis sollten Sie nun im Rechenblatt sehen, daß das Feld D5 den Text "Hallo Welt" erhalten hat.

Der Anfang eines Macros kann an einer beliebigen Stelle des Macro-Blattes stehen. Die nachfolgenden Anweisungen müssen sich aber jeweils genau im nächsten Feld darunter befinden. Der RETURN()-Befehl bildet das Ende des Macros und übergibt die Kontrolle wieder dem Benutzer.

Die direkte Eingabe von Macros ist mühsamer und zeitintensiver als das einfache "Aufzeichnen". Außerdem werden - besonders bei komplexen Macros - Fehler immer wahrscheinlicher. Die Fehlersuche in einem Macro ist dann eine sehr undankbare Aufgabe, vor allem wenn man anderes zu tun hat. Wichtig ist jedoch die Möglichkeit, aufgezeichnete Macros 'von Hand' weiter zu bearbeiten. Alle Macro-Befehle, welche K-Spread beim Aufzeichnen erzeugt, finden Sie im Nachschlageteil dieses Kapitels.

Abbrechen einer Macro-Ausführung

Wenn Sie das Ausführen eines Macros abbrechen wollen, drücken Sie einfach die **Shift**-Taste. Nach dem momentanen Macro-Befehl wird gestoppt.

Das MACRO-Menü

Macro anlegen..

Sie werden aufgefordert, ein Macro festzulegen, indem Sie einen Namen eingeben und den Feldbereich angeben, in dem die Macro-Befehle stehen. Es muß dazu das entsprechende Macro-Blatt im obersten Fenster dargestellt sein.

Macro umbenennen..

ermöglicht es Ihnen, Name und Lage eines Macros zu ändern, die Sie bei "Macro anlegen.." festgelegt haben. Auch hier muß das oberste Fenster das .SPM-Fenster sein.

Macro löschen..

In der bekannten Auswahlbox können Sie das Macro bestimmen, das anschließend aus der Macro-Liste entfernt wird. Das entsprechende Macro-Fenster muß aktiviert sein.

Macroanfang setzen

Wenn Sie sich im .SPM-Fenster befinden und diesen Menüpunkt anklicken, wird das Feld, auf dem der Edit-Rahmen steht, zum Anfangsfeld des Macros. Wählen Sie aus einem .SPD-Fenster den Aktionen-Recorder mittels "Aufnahme starten" an, so beginnt die Ausgabe der Macro-Befehle ab diesem Anfangsfeld.

Macrotaste..

Macros können Sie auch schnell über die Tastatur aufrufen. Sie müssen nur vorher festlegen, welches Macro welcher Taste zugeordnet werden soll. Dies erledigen Sie in diesem Menüpunkt. Zuerst bestimmen Sie in der Auswahlbox das gewünschte Macro (das Macro-Fenster muß ganz oben sein). Anschließend werden sie aufgefordert, eine Zahl zwischen 1 und 10 eingeben, welche die zugehörige Funktionstaste F1 bis F10 bestimmt. (Geben Sie eine 0 ein, so wird die Zuordnung des Macros zu einer Funktionstaste aufgehoben.)

Um zu einem beliebigen späteren Zeitpunkt das Macro zu starten, betätigen Sie einfach die zugehörige Funktionstaste, während Sie ALTERNATE gedrückt halten.

Immer Neuzeichnen

legt fest, ob ein Neuzeichnen der Fenster nach jedem einzelnen Macro-Befehl stattfinden soll, oder erst nachdem das ganze Macro beendet ist.

Macro ausführen..

In der Macro-Auswahlbox können Sie das Macro angeben, das anschließend gestartet wird.

Timer..

Mit Hilfe eines sogenannten "Timers" ist es möglich, daß an einem bestimmten Datum zu einer festgelegten Uhrzeit ein Macro automatisch gestartet wird. Sie können sich so beispielsweise einen Terminplaner in K-Spread 4 programmieren.

Nach dem Anklicken des Menüpunktes "Timer.." erscheint die gewohnte Auswahlleiste, mit der Sie eine Timer-Angabe neueingeben, ändern, löschen oder alle entfernen wollen.

Neu:

Das Programm verlangt die Eingabe des Startdatums und anschließend der Startzeit (beispielsweise "12/11/91" und "10:12"). Sie müssen nun lediglich noch in der Auswahlbox festlegen, welches Macro zu diesem Zeitpunkt ausgeführt werden soll.

ändern:

In einer Auswahlbox können Sie einen aufgelisteten Zeitpunkt bestimmen. Anschließend finden dieselben Abfragen wie bei Timer - Neu statt.

löschen:

Mittels der Auswahlbox legen Sie fest, welcher zeitgesteuerte Macroaufruf entfernt werden soll.

Reset:

Es erscheint eine Sicherheitsabfrage, ob Sie auch wirklich ALLE zeitabhängigen Ausführungen von Macros löschen wollen.

Aufnahme starten/Aufnahme beenden

Wenn Sie den Aktionen-Recorder benutzen, können Sie so von einem aktiven Rechenblatt-Fenster aus die Aufnahme starten und beenden. Die eingetragenen Macro-Befehle sehen Sie anschließend im Macro-Fenster.

Zum Aufbau der Macro-Befehle

Ein Macro-Kommando ist im Grunde nichts anderes als eine spezielle Funktion. In der Schreibweise stimmen Macro-Befehle und die eingebauten Funktionen im wesentlichen überein. Die Unterschiede oder nochmals zu erwähnende Punkte werden im folgenden aufgezeigt:

Optionale Parameter

Parameter, welche im Nachschlageteil zu den Macro-Befehlen in geschweifte Klammern {} eingeschlossen sind, können im Macro-Kommando weggelassen werden. Es werden dann automatisch festgelegte Grundwerte eingesetzt, sogenannte Default-Werte. Diese werden beim jeweiligen Kommando aufgelistet. Sind solche Defaults nicht angegeben, so wird bei einem Weglassen eines optionalen Parameters an der ursprünglichen Einstellung nichts geändert. Wenn die Angabe eines Bereichs oder Feldes einen optionalen Parameter darstellt, so führt ein Fehlen dieser Angabe dazu, daß der markierte Bereich im Rechenblatt eingesetzt wird.

Auch wenn Sie einen Parameter weglassen, müssen Sie trotzdem trennende Kommas eingeben, wenn mehr als ein Parameter verlangt wird. Beispiel: Der Macro-Befehl besitze folgende Syntax, d.h Schreibweise:

=KOMMANDO({Parameter1}, {Parameter2}, {Parameter3})

Wenn Sie nun manche Parameter nicht angeben wollen, könnte Ihre Eingabe folgendermaßen aussehen:

=KOMMANDO(,Parameter2,) oder

=KOMMANDO(,,Parameter3)

Die Funktionen SELECT(), FORMULA(), REFTEXT() und TEXTREF() unterscheiden sich von den anderen dadurch, daß hier die trennenden Kommas nicht benötigt werden.

A0_Koordinaten

werden in einem Rechenblatt dazu benutzt, einzelne Felder und Bereiche anzusprechen, z.B. A5, A5, [A5]. Diese Angaben bezeichnen beim Ablauf eines Macros Felder, die unabhängig von der Position der Cursor sind. Sie werden deshalb verwendet, um auf Konstanten zu verweisen oder um den Edit-Rahmen beim Start eines Macros zu setzen.

RC_Koordinaten

sind Koordinaten, die sich auf die letzte Position des Cursors beziehen. "R" steht hierbei für Zeilen (= **R**ows), "C" für Spalten (= **C**olumns). Der Cursor wird also relativ zu sich selbst bewegt. RC_Koordinaten werden immer als Zeichenketten angegeben, um sie von den A0_Koordinaten zu unterscheiden. Die Schreibweise "R[2]C[-4]" bezieht sich auf den Edit-Rahmen, "R[0]C[0]:R[4]C[7]" auf den Feld-Marker. Positive Zahlen in den eckigen Klammern repräsentieren entweder die Anzahl Zeilen unter dem Cursor oder die Anzahl Spalten rechts des Cursors; negative Zahlen entsprechend die Zeilen oberhalb oder Spalten links des Cursors. Als Position des Feld-Markers wird die linke obere Ecke eines selektierten Bereichs angesehen.

Da RC_Koordinaten stets als Zeichenketten übergeben werden, findet eine Überprüfung, ob sich das angesprochene Feld noch innerhalb des Rechenblatts befindet, erst während des Macroablaufs statt. Liegt das Feld außerhalb des Fensterbereichs, erscheint eine Dialogbox, welche den Fehler meldet. Der fehlerhafte Macro-Befehl wird in jedem Fall nicht bearbeitet. Sie können auswählen, ob Sie mit der Ausführung des Macros fortfahren möchten. Dies kann Sie auf weitere unerlaubte Feldverweise führen.

Beachten Sie bitte, daß A0 und RC_Koordinaten auch Rechenblatt-Namen enthalten können, z.B. STEUER91.SPD!A0. Sie haben also auch Zugriff auf andere Rechenblätter im Speicher oder auf Diskette.

Eingabe von Zahlen in ein Macro-Blatt

Es gibt einige Einschränkungen was die Eingabe von Daten in ein Macro-Blatt betrifft, weil das Neuberechnen abgeschaltet ist. Zahlen und Formeln, welche direkt eingegeben werden, enthalten keine brauchbaren Werte, bis das Macro ausgeführt wird, in dem sie liegen. Das bedeutet wiederum, daß Zahlen und Formeln, die nicht innerhalb eines Macros liegen, keine sinnvollen Werten haben können.

Boolsche Variablen

Es gibt Funktionen, welche als Ergebnis nur TRUE (wahr) oder FALSE (falsch) enthalten können. Ein Inhalt 0 bedeutet FALSE, jeder andere Wert TRUE.

Zum Nachschlageteil der Macro-Befehle

Für die folgende Auflistung aller Macro-Befehle treffen wir einige Vereinbarungen, damit eine einheitliche Schreibweise möglich ist:

A0_Feld	:	A0_Koordinaten eines Feldes
RC_Feld	:	RC_Koordinaten " "
A0_Bereich	:	A0_Koordinaten eines Bereichs
RC_Bereich	:	RC_Koordinate " "
/	:	oder
"..."	:	eine Zeichenkette muß in Anführungszeichen gesetzt werden
{...}	:	das in gechweifte Klammern Gesetzte kann weggelassen werden; es wird dann automatisch der Default-Wert eingesetzt.

Anmerkung zu den BLATT-Menü-Macros

Für alle Macro-Befehle zum BLATT-Menü (TEXT_COLOUR(), BACK_COLOUR, BACK_PATTERN, HJUSTIFY, CELL_LEAD, CELL_TRAIL, CELL_WRAP, CELL_COMMA, CELL_UNDERLINE, CELL_DIVIDER, CELL_FONT und CELL_FORMAT()) gilt, daß die einzelnen Festlegungen jeweils als Grundeinstellung (="Default") oder zum sofortigen Ändern von Feldern verwendet werden können.

Wenn Sie nur ein Default setzen, sind davon schon belegte Felder NICHT betroffen, sondern erst Felder, die Sie anschließend eingeben.

Zudem gelten für die genannten Funktionen die Beschreibungen zu A0_Bereich/"RC_Bereich"/A0_Feld/"RC_Feld" und Typ, die beim Macro-Kommando TEXT_COLOUR() vorgenommen werden. Es wird darauf verzichtet, jedesmal auf TEXT_COLOUR() zu verweisen.

Anmerkung zu den DATEI-Menü-Macros

Die entsprechend markierten Funktionen haben dieselbe Wirkung wie das Anklicken der entsprechenden Menüpunkte im DATEI-Menü. Eine ausführliche Erklärung der Funktionen finden Sie bei der Beschreibung des DATEI-Menüs.

Einigen dieser Funktionen kann ein Dateiname übergeben werden. Dieser taucht bei der Angabe der korrekten Syntax als "Dateiname" auf. Um die Erkärung nicht jedesmal überflüssigerweise zu geben, sei sie hier aufgeführt. **Dateiname:** Wenn sich die Datei im aktuellen Directory befindet, genügt die Angabe des Dateinamens. Ansonsten muß der vollständige Pfad übergeben werden.

Wenn "Dateiname" in geschweiften Klammern ({"Dateiname"}) aufgeführt ist, dürfen Sie diesen Parameter weglassen. Es erscheint dann die Dateiauswahlbox, in der Sie die gewünschte Datei bestimmen können.

Anmerkung zu den Macros für erweiterte Dateihandhabung

Die mit *(Siehe Anm. zu Macros für erw. Dateihandhabung")* gekennzeichneten Macro-Funktionen werden vielleicht nicht von den meisten Anwendern von K-Spread benutzt. Wenn Sie aber einige Erfahrung im Programmieren von Macros haben, so können Sie sich mit ihnen die ganzen Möglichkeiten von K-Spread 4 zunutze machen.

Diese Macro-Funktionen dienen dazu, Dateien auf unterschiedliche Weise zu lesen oder zu schreiben. Die Befehle können Sie benutzen, um ein Macro zu erzeugen, das beispielsweise ein ganz individuelles Dateiformat handhaben kann oder das nur ganz bestimmte Zeichen einliest. Es ist Ihnen somit möglich, ein Macro-Programm zu schreiben, das Dateien der unterschiedlichsten Formate verarbeiten kann. Sie müssen lediglich die Regeln kennen und verstehen, nach denen sich ein bestimmter Dateityp zusammensetzt.

Beachten Sie bitte folgendes: Dateien werden im ASCII-Modus geöffnet. Das Zeichen mit dem ASCII-Code 0 wird als das Ende einer Zeichenkette aufgefaßt. Es kann also vorkommen, daß beispielsweise LENGTH() andere Zahlen zurückliefert, als Sie denken, daß eingelesen wurden.

WARNUNG: Diese Funktionen verarbeiten Daten auf sehr niedriger Stufe. Sie müssen somit sicherstellen, daß keine Fehler auftreten. Schon kleine Fehler können zu einem großen Datenverlust führen. Setzen Sie diese mächtigen Möglichkeiten, die K-Spread Ihnen bietet, deshalb besonders verantwortungsbewußt ein.

Anmerkung zu den Datenbank-Macros

Vergleichen Sie bitte unbedingt die Erläuterungen des Kapitels "Das DATENBANK-Menü".

Die Macro-Befehle

ABSREF

liefert die absolute Bezeichnung eines Feldes oder Bereichs. Als Parameter werden die RC_Koordinaten relativ zu einem bestimmten Feld oder Bereich verlangt.

Syntax: ABSREF("RC_Offset", A0_Feld/A0_Bereich)
RC_Offset: Distanz zu den übergebenen A0_Koordinaten
z.B. ABSREF("R[-1]C[-1]", G6) ergibt F5
ABSREF("R[1]C[1]", G6) ergibt H7
ABSREF("R[1]C[1]", G6:H7) ergibt H7:I8

ACTIVE_CELL()

liefert die Bezeichnung des Feldes zurück, auf dem der Edit-Rahmen steht.

Syntax: ACTIVE_CELL()

z.B. ACTIVE_CELL() liefert A2, wenn sich dort der Edit-Rahmen befindet.

Diese Funktion läßt sich sehr gut mit ABSREF() benutzen: ABSREF("R[2]C[1]", ACTIVE_CELL()) ergibt E6, falls auf D4 der Edit-Rahmen steht.

ACTIVE_RNG()

liefert die Bezeichnung des Bereichs, der durch den Feld-Marker gekennzeichnet wird.

Syntax: ACTIVE_RNG()

z.B. ACTIVE_RNG() liefert B2:D4, falls sich dort der Feld-Marker befindet.

Wenn Sie vor Aufruf des Macros einen Bereich selektiert haben, so können Sie mit dieser Funktion diesen Bereich im Macro in Erfahrung bringen. Dies stellt eine mögliche Form der "Eingabe" dar.

AUTO_CALC

schaltet das automatische Neuberechnen ein oder aus.

Syntax: AUTO_CALC({einschalten})
einschalten: TRUE: Ja (Default); FALSE: Nein.

BACK_COLOUR (siehe Anmerkung zu den BLATT-Menü-Macros) erlaubt Ihnen, die Hintergrundfarben festzulegen.

Syntax: BACK_COLOUR({A0_Bereich/"RC_Bereich"/A0_Feld/" RC_Feld"}, {Typ}, {Farbe})

Es gelten die Erklärungen, die bei TEXT_COLOUR() gemacht wurden.

BACK_PATTERN (siehe Anmerkung zu den BLATT-Menü-Macros) bestimmt das Muster im Hintergrund der Felder.

Syntax: BACK_PATTERN({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"}, {Typ}, {Muster})

Muster: Füllmuster, welches bestimmte Felder erhalten sollen.

CELL_COMMA (siehe Anmerkung zu den BLATT-Menü-Macros) legt fest, ob in Zahlen Kommas zur Tausendertrennung verwendet werden sollen.

Syntax: CELL_COMMA({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"}, {Typ}, {Komma})

Komma: TRUE: ja; FALSE: nein

CELL_DIVIDER (siehe Anmerkung zu den BLATT-Menü-Macros)

Syntax: CELL_DIVIDER({A0_Bereich/"RC_Bereich"/A0_Feld/" RC_Feld"}, {Typ}, {ASCII_Wert_Trennzeichen})

CELL_FIND

sucht das nächste Auftreten eines angebbaren Feldinhalts.

Syntax: CELL_FIND("Text"/Zahl)

z.B. CELL_FIND("Gehalt") sucht spaltenweise das nächste Feld, das den Text "Gehalt" enthält. Das gefundende Feld wird zum aktivem Feld, d.h. der Edit-Rahmen darauf gesetzt.

CELL_FONT (siehe Anmerkung zu den BLATT-Menü-Macros) legt die Attribute fest, anhand derer Texte in Feldern ausgegeben werden.

Syntax: CELL_FONT({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"}, {Typ}, {Attribute})

Attribute: Legen das Aussehen einer Zeichenkette fest. Um sie beliebig zu kombinieren, addieren Sie einfach die entsprechenden Werte der gewünschten Attribute:

1: fett
2: hell
4: schräg
8: unterstrichen
16: umrandet

CELL_FORMAT (siehe Anmerkung zu den BLATT-Menü-Macros) bestimmt das Anzeigeformat eines Zahlenwertes.

Syntax: CELL_FORMAT({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"}, {Typ}, {Format}, {Größe})

Format:		Größe:
0	- Kommazahl	Nachkommastellen (0..14)
1	- Exponentialschreibweise	Nachkommastellen (0..14)
2	- Prozentschreibweise	Nachkommastellen (0..14)
3	- Datum	Anzeigeart (0..14)
4	- Uhrzeit	Anzeigeart (0..14)
5	- zu bestimmter Basis	(1..16, 1 = Römische Ziffern, 16 = arabisch)
6	- Text (nur bei Formeln)	(unbenutzt)
7	- versteckt	(unbenutzt)

CELL_LEAD (siehe Anmerkung zu den BLATT-Menü-Macros) setzt kurze Zeichenketten, die den Feldinhalten vorangestellt werden.

Syntax: CELL_LEAD({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"}, {Typ}, {Prefix})

Prefix: Zeichenkette, die vorangestellt wird:

0 - keine

1, 2, oder 3 - 1., 2. oder 3. Zeichenkette, die durch LEADING() oder "Prefix/Suffix.." im BLATT-Menü festgelegt wurden.

CELL_PROTECT (siehe Anmerkung zu den BLATT-Menü-Macros) schützt Felder gegen Überschreiben oder hebt diesen Schutz wieder auf.

Syntax: CELL_PROTECT({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"}, {Typ}, {schützen})

schützen: Schaltet den Schutz; TRUE: ein; FALSE: aus.

CELL_REPLACE

sucht einen bestimmten Feldinhalt und ersetzt diesen, sobald er gefunden wird. Es können nur Feldinhalte gleichen Datentyps ersetzt werden, d.h. Zahlen durch Zahlen und Texte mit Texten.

Syntax: CELL_REPLACE ({"Suchen-Text"/Suchen-Zahl}, {"Ersetzen-Text"/Ersetzen-Zahl})

z.B. CELL_REPLACE("Gehalt", "Lohn") ersetzt im nächsten gefundenen Feld die Zeichenkette "Gehalt" durch "Lohn".

Wenn Sie gesuchten Inhalt aller Felder, die sich ein einem bestimmten Feldbereich befinden, ersetzen wollen, müssen Sie "Schleifen" verwenden. Schauen Sie dazu bei IF und GOTO nach.

CELL_TRAIL (siehe Anmerkung zu den BLATT-Menü-Macros) bestimmt die Zeichenketten, die direkt hinter den Feldinhalten ausgegeben werden.

Syntax: CELL_TRAIL({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"}, {Typ}, {Suffix})

Es gilt entsprechend das soeben bei CELL_LEADING() Erwähnte.

CELL_UNDERLINE (siehe Anmerkung zu den BLATT-Menü-Macros) bestimmt, ob Felder unterstrichen werden sollen.

Syntax: CELL_UNDERLINE({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"}, {Typ}, {unterstreichen})

unterstreichen: TRUE: Ja; FALSE: Nein

CELL_WRAP(siehe Anmerkung zu den BLATT-Menü-Macros) schaltet den Zeilenumbruch in Textfeldern ein oder aus.

Syntax: CELL_WRAP({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"}, {Typ}, {Umbruch})

Umbruch: TRUE: Zeilenumbruch ein; FALSE: aus

CHANGE_TYPES

legt den Formeltyp fest.

Syntax: CHANGE_TYPES({A0_Bereich/"RC_Bereich"}, {Formel -> Text}, {Formel -> Zahl}, {bewahre_Format})

A0_Bereich/"RC_Bereich": Bereich von Feldern, die konvertiert werden sollen. Es sind nur Felder von der Umwandlung betroffen, die Formeln enthalten.

Formel -> Text: Formeln, die Zeichenketten als Ergebnis liefern, werden in Text umgewandelt. TRUE: ja (Default), FALSE: nein.

Formel -> Zahl: Formeln, die Zahlen zurückliefern, werden in Zahlen umgewandelt.

TRUE:ja (Default), FALSE: Nein.

bewahre_Format: soll das Anzeigeformat beibehalten oder den Einstellungen angepaßt werden? TRUE: Beibehalten (Default), FALSE: Anpassen.

z.B. CHANGE_TYPES(A0:M20, 1,0,) ändert Text, aber keine Zahlen.

CLEAR()

löscht ALLE Daten des aktuellen Blattes. **Vorsicht!**

Syntax: CLEAR()

COL_WIDTH

legt die Spaltenbreite einer Spalte oder eines ganzen Spaltenbereichs fest.

Syntax: COL_WIDTH({A0_Bereich/"RC_Bereich"}, {Breite})

A0_Bereich/"RC_Bereich": der betroffene Bereich muß vollständig übergeben werden, z.B. A0:C0 oder "RC[0]:RC[4]". Wenn diese Angabe weggelassen wird, definieren die markierten Felder den Bereich.

Breite: 0 - versteckt die Spalten

3..80 - soviele Zeichen beträgt die neue Spaltenbreite

Wenn dieser Parameter fehlt, wird der aktuelle Wert der Grundeinstellung der Spaltenbreite (festgelegt durch das letzte Anwählen von "ALLE" in der Dialogbox zur Spaltenbreite) herangezogen.

COMBINE

lädt einen Bereich von Diskette in das aktuelle Rechenblatt ein.

Syntax: COMBINE({"Dateiname"}, A0_Bereich, {"RC_Bereich"}, {Typ})

Dateiname: Name der Datei, welche geladen werden soll.

A0_Bereich : der Bereich in der Datei, der geladen werden soll.

RC_Bereich: der Bereich des aktuellen Rechenblatts, in den der Bereich geladen werden soll. Wenn Sie nichts übergeben, sind nur die augenblicklich markierten Felder betroffen.

Typ: 0 = ERSETZEN (Default)
1 = ADDIEREN
2 = ABZIEHEN

z.B. COMBINE("TEST.SPD", BereichName, A0: A5, 1)
COMBINE(, A0: A7,,)

CONTROL_CODES

sendet Steuerzeichen an den Drucker oder schreibt sie in Felder.

Syntax: CONTROL_CODES({"Device"/A0_Bereich/"RC_Bereich"}, "Steuerzeichen")

Device: legt fest, wohin die Ausgabe geleitet werden soll:
"PRN:" - Parallelport
"AUX:" - Serieller Port

Bereich: Es kann anstelle eines Devices ein Bereich angegeben werden. Die Steuerzeichen werden dann in diese Felder geschrieben, und erst an den Drucker ausgegeben, wenn diese Felder gedruckt werden.

Steuerzeichen: Zeichenkette, die alle zu sendenden Zeichen enthält.

z.B. CONTROL_CODES("PRN:", " ""A"" 16") sendet ein A, gefolgt von dem ASCII-Zeichen mit dem Wert 16.

COPY

kopiert Felder von einer Stelle auf eine andere.

Syntax: COPY({A0_Bereich1/"RC_Bereich1"/A0_Feld1/"RC_Feld1"}, {A0_Bereich2/"RC_Bereich2"/A0_Feld2/"RC_Feld2"}, {ausfüllen}, {Ergebnisse})

(Bereich1/Feld1): Quellbereich/-feld

(Bereich2/Feld2): Zielbereich/-feld

ausfüllen:	TRUE	- der Quellbereich wird solange kopiert, bis der Zielbereich gefüllt ist
	FALSE	- nicht voll ausfüllen (Default)
Ergebnisse:	TRUE	- nur die Ergebnisse von Formeln, nicht jedoch die Formeln selbst kopieren
	FALSE	- Ergebnisse UND Formeln kopieren

z.B. COPY(A0:B5, C6:D11, 0, 0)

CREATE_DSET

legt ein Datenblatt, d.h. entweder ein Rechenblatt oder ein Macroblatt, an.

Syntax: CREATE_DSET("Name", Typ)

Name: Bezeichnung, die das Datenblatt erhalten soll. Die Endung .SPD oder .SPM wird je nach Typ angefügt.

Typ: = 0: Rechenblatt

= 1: Macroblatt

z.B. CREATE_DSET("Test", 0) erzeugt ein Rechenblatt "TEST.SPD"

CREATE_DSET("Test", 1) erzeugt ein Macroblatt "TEST.SPM"

CREATE_RNAME

weist einem Bereich einen Namen zu.

Syntax: CREATE_RNAME("Name", A0_Bereich/"RC_Bereich")

z.B. CREATE_RNAME("Einnahmen", A0:B6)

CREATE_TRACE

legt eine neue Überwachung eines Feldes oder Bereichs fest.

Syntax: CREATE_TRACE({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"}, Typ, {Wert})

Typ: Art der Überprüfung:

0 - unterschreiten des Wertes

1 - übersteigen
2 - erhalten
3 - verändern

Wert: Zahl, mit welcher der Feldinhalt verglichen wird.

z.B. CREATE_TRACE(A0, 0, 12) überwacht das Feld A0 und meldet, wenn dieses den Wert 12 unterschreitet.

Beachten Sie, daß bei der Überwachung eines Bereichs für jedes einzelne Feld eine Überwachung festgelegt wird.

CREATE_UDEF

dient zum Erstellen einer Neuen Funktion.

Syntax: CREATE_UDEF("Name", Parameter, "Formel", {global})

Name: Name, mit welchem die neue Funktion angesprochen werden soll

Parameter: Anzahl der Parameter: Zahl von 0..9

Formel: Rechenanweisung, in der die Parameter als %1, %2, .., %9 bezeichnet werden.

global: TRUE: Die neue Funktion ist in jedem Blatt bekannt
FALSE: Nur im aktuellen Blatt ist die Neue Funktion verwendbar (Default)

z.B. CREATE_UDEF("Zins", 1, "%1*0.10", 1). Ein Aufruf mit ZINS(100) liefert 10 zurück.

DBASE_CRITERIA

bestimmt den Bereich innerhalb einer Datenbank (einschließlich der Überschriften), in dem sich die Kriterien befinden.

Syntax: DBASE_CRITERIA({A0_Bereich/"RC_Bereich"})

z.B. DBASE_CRITERIA(C0:C1)

DBASE_DELETE()

sucht ALLE zu den Kriterien passenden Datenbank-Felder und löscht diese dann.

Syntax: DBASE_DELETE()

DBASE_DELETE_RECORDS

löscht die Datensätze im angegebenen Datenbank-Bereich.

Syntax: DBASE_DELETE_RECORDS({A0_Bereich/"RC_Bereich"})

z.B. Wenn der Datenbankbereich A0:B100 ist, so löscht DBASE_DELETE_RECORDS(A3:A5) die Feldinhalte von A3:B3, A4:B4, A5:B5.

DBASE_EXTRACT

sucht die zu den Kriterien passenden Datenbank-Felder und gibt diese in den Ausgabebereich.

Syntax: DBASE_EXTRACT({ohne_Duplikate})

ohne_Duplikate: FALSE - Sie wollen ALLE passenden Einträge, auch wenn sie doppelt sind. (Default)

TRUE - Ein passender Eintrag, der schon einmal gefunden wurde, wird nicht beachtet.

DBASE_FIND

durchsucht den ganzen Datenbankbereich nach den aktuellen Kriterien. Wenn die Suche erfolgreich ist, wird auf das erste gefundene Feld der Edit-Rahmen gesetzt. Selbstverständlich geht diese Funktion (wie auch DBASE_EXTRACT und DBASE_DELETE) davon aus, daß die entsprechenden Bereiche der Datenbank bereits angegeben wurden.

Syntax: DBASE_FIND({vom_Anfang})

vom_Anfang: TRUE - beginnt die Suche am Anfang der Datenbank (Default)

FALSE - setzt Suche nach dem letzten gefundenen Feld fort

z.B. DBASE_FIND(0) sucht nach der ersten Übereinstimmung

DBASE_FIND(1) sucht nach dem nächsten Treffer

DBASE_FIND(1) sucht nach dem nächsten Treffer...

DBASE_INSERT_RECORDS

schafft Platz in der Datenbank, damit Sie neue Datensätze einfügen können.

Syntax: DBASE_INSERT_RECORDS({A0_Bereich/"RC_Bereich"})

A0_Bereich/"RC_Bereich": Die Datensätze, welche sich in diesem Bereich befinden, werden nach unten verschoben. Anschließend wird der Bereich gelöscht und steht für Neueingaben zur Verfügung.

z.B. Wenn der Datenbankbereich A0:B100 ist, so schafft DBASE_INSERT_RECORDS(A3:A5) Platz in A3:B3, A4:B4, A5:B5.

DBASE_OUTPUT

legt den Datenbank-Bereich fest, in den die Ausgaben vorgenommen werden.

Syntax: DBASE_OUTPUT({A0_Bereich/"RC_Bereich"})

A0_Bereich/"RC_Bereich": Ausgabebereich inklusive der Überschriften. Wenn der Bereich nur aus den Überschriften besteht, benutzt das Programm alle benötigten Felder, die unterhalb davon liegen. Setzt sich der angegebene Bereich nicht nur aus den Überschriften zusammen, werden auch nur die Felder ohne Überschriften benutzt.

DBASE_RANGE

legt den gesamten Feldbereich (inklusive der Bezeichnungen) fest, der für die Datenbank verwendet wird.

Syntax: DBASE_RANGE({A0_Bereich/"RC_Bereich"})

A0_Bereich/"RC_Bereich": Wie zu Anfang erwähnt, gilt auch hier: fehlt dieser Parameter, werden die durch den Feld-Marker selektierten Felder verwendet.

z.B. DBASE_RANGE(A0:D100)

DBASE_RESET()

setzt den Datenbankbereich, den Kriterienbereich und den Ausgabebereich auf A0:A0. Dies bewirkt, daß die Datenbankfunktionen nicht mehr benutzt werden können. Sie müssen zuerst wieder die genannten Bereiche festlegen.

Syntax: DBASE_RESET()

DBASE_INSERT_RECORDS

schafft Platz in der Datenbank, damit Sie neue Datensätze einfügen können.

Syntax: DBASE_INSERT_RECORDS({A0_Bereich/"RC_Bereich"})

A0_Bereich/"RC_Bereich": Die Datensätze, welche sich in diesem Bereich befinden, werden nach unten verschoben. Anschließend wird der Bereich gelöscht und steht für Neueingaben zur Verfügung.

z.B. Wenn der Datenbankbereich A0:B100 ist, so schafft DBASE_INSERT_RECORDS(A3:A5) Platz in A3:B3, A4:B4, A5:B5.

DBASE_DELETE_RECORDS

löscht die Datensätze im angegebenen Bereich.

Syntax: DBASE_DELETE_RECORDS({A0_Bereich/"RC_Bereich"})

z.B. Wenn der Datenbankbereich A0:B100 ist, so löscht DBASE_DELETE_RECORDS(A3:A5) die Feldinhalte von A3:B3, A4:B4, A5:B5.

DELETE

löscht einen Feldbereich aus dem aktuellen Rechenblatt.

Syntax: DELETE({A0_Bereich/"RC_Bereich"})

DELETE_COLS

löscht Spalten aus dem Rechenblatt.

Syntax: DELETE_COLS("Spalte", Anzahl)

z.B. DELETE_COLS("B", 2) löscht die Spalten "B" und "C".

DELETE_DSET

entfernt ein bestehendes Datenblatt aus dem Speicher.

Syntax: DELETE_DSET("Name")

Name: Name des zu löschenden Blattes

DELETE_RNAME

entfernt einen Bereichsnamen aus dem Speicher.

Syntax: DELETE_RNAME("Name")

z.B. DELETE_RNAME("Zuschüsse")

DELETE_ROWS

löscht Zeilen aus dem Rechenblatt.

Syntax: DELETE_ROWS(Zeile, Anzahl)

Es gilt das bei DELETE_COLS() Beschriebene.

DELETE_TRACE

entfernt die Überwachung eines Feldes oder Bereichs aus dem Speicher.

Syntax: DELETE_TRACE({A0_Bereich/"RC_Bereich"/A0_Feld/ "RC_Feld"})

DELETE_UDEF

entfernt eine Neue Funktion aus dem Speicher.

Syntax: DELETE_UDEF("Name", {global})

global: die zu löschende Neue Funktion
TRUE: Befindet sich in der globalen Liste.
FALSE: Gehört nur zum aktuellen Rechenblatt. (Default)

z.B. DELETE_UDEF("Zins", 1) löscht die neue Funktion ZINS(), welche in allen Blättern verwendet werden kann.

DROP_MENUS

erlaubt das Ein- und Ausschalten der Drop-Down-Menüs, die erscheinen, wenn Sie auf Felder, Bereiche oder Spalten-/Zeilenköpfe klicken.

Syntax: DROP_MENUS({anschalten})

anschalten: TRUE - Drop-Down-Menüs erscheinen
FALSE - Menüs erscheinen nicht (Default)

EDIT_RNAME

ändert den Namen eines Bereichs und den Feldbereich selbst.

Syntax: EDIT_RNAME("alter_Name", "neuer_Name", A0_Bereich/ "RC_Bereich")

A0_Bereich/"RC_Bereich": Neue Lage und Ausdehnung des Bereichs

z.B. EDIT("Einnahmen", "Zuschüsse", A0:B3)

FDELETE

löscht eine beliebige Datei von Diskette.

Syntax: FDELETE({"Dateiname"})

z.B. FDELETE("TEST.DOC")

FILL

füllt einen Bereich mit einer Folge von Zahlen.

Syntax: FILL({A0_Bereich/"RC_Ber."}, Start, Summand, zeilenweise)

Start: Zahlenwert des linken oberen Feldes des Bereichs

Summand: Zahlenwert, der von Feld zu Feld hinzuaddiert wird

zeilenweise: die Felder werden gefüllt TRUE: nach Zeilen; FALSE: nach Spalten

z.B. FILL(A0:B1, 2, 3, 0) ergibt A0=2, A1=5, B0=8, B1=11.

FORM_FEED

veranlaßt den Drucker zu einem Seitenvorschub.

Syntax: FORM_FEED({Anzahl})

Anzahl: der auszulösenden Seitenvorschübe. (Default ist 1)

FORMULA

übergibt neue Inhalte in ein Feld des Rechenblattes.

Syntax: FORMULA(Zahl/"Text"/"=Formel", {A0_Feld/"RC_Feld"})

Zahl: Diese Zahl wird in das Feld geschrieben

Text: Dieser Text wird in das Feld geschrieben.

Formel: Diese Formel (mit einem "=" am Anfang) wird übergeben

Wenn die Angabe A0_Feld/"RC_Feld" fehlt, wird demjenigen Feld ein neuer Inhalt übergeben, auf welchem der Edit-Rahmen steht. Ist die Angabe jedoch vorhanden, so wird vor der Übergabe des neuen Feldinhalts der Edit-Rahmen auf das angesprochene Feld gesetzt, genau wie bei SELECT().

z.B. FORMULA(100) schreibt in das derzeit aktive Feld die Zahl 100
FORMULA("Text", A0) schreibt in das Feld A0 den Text "Text"
FORMULA("=A0*20", A1) - " - A1 die Formel "=A0*20"

FREAD (Siehe "Anm. zu Macros für erw. Dateihandhabung")

liest n Zeichen (Bytes) in ein anzugebendes Feld.

Syntax: FREAD(Anzahl, A0_Feld/"RC_Feld")

Anzahl: Anzahl der zu lesenden Bytes

Feld: Bezeichnung des Feldes, das die gelesenen Daten als Feldinhalt zugewiesen bekommt.

FREADLN (Siehe "Anm. zu Macros für erw. Dateihandhabung")

liest eine Zeichenkette in ein Feld.

Syntax: FREADLN(A0_Feld/"RC_Feld")

Feld: Feld, das die Zeichenkette als Inhalt erhält.

FREQ_DIST

bestimmt die Häufigkeitsverteilung bestimmter Werte.

Syntax: FREQ_DIST({A0_Bereich1/"RC_Bereich1"},
A0_Bereich2/"RC_Bereich2")

A0_Bereich1/"RC_Bereich1": Bereich, in dem die Werte stehen, deren Häufigkeit ermittelt wird.
A0_Bereich2/"RC_Bereich2": Ausgabebereich
z.B. FREQ_DIST(A0: A20, C0: D5)

FSEEK (Siehe "Anm. zu Macros für erw. Dateihandhabung")
setzt den "Dateizeiger" (Filepointer). Dieser Filepointer zeigt auf ein Zeichen in der Datei und kann mit dieser Funktion an eine andere Stelle gesetzt bzw. relativ zu seiner derzeitigen Position verschoben werden.

Syntax:	FSEEK(Lese_Schreibe, Offset, Offsetbezug)
Lese_Schreibe:	0 - lese Datei
	1 - schreibe Datei
Offset:	Abstand vom Offsetbezug in Bytes
Offsetbezug:	0 - Dateianfang
	1 - aktuelle Position des Filepointers
	2 - Dateiende
z.B.	FSEEK(0, 1, 1) bewegt den Zeiger ein Zeichen weiter.

FTELL (Siehe "Anm. zu Macros für erw. Dateihandhabung")
liefert die aktuelle Position des Filepointers zurück.
Syntax: FTELL(Zahl)

FWRITE (Siehe "Anm. zu Macros für erw. Dateihandhabung")
schreibt eine Zeichenkette ohne ASCII-0 am Ende in eine Datei.
Syntax: FWRITE("Zeichenkette")
Zeichenkette: Daten, die als ASCII-Zeichen aufgefaßt werden. Die Zeichenkette ist nicht mit ASCII 0 abgeschlossen.

FWRITELN (Siehe "Anm. zu Macros für erw. Dateihandhabung")
schreibt den Inhalt eines anzugebenden Feldes in eine Datei, und fügt am Ende ein ASCII-0-Zeichen zur Zeilenendekennung an.
Syntax: FWRITELN("Zeichenkette")
Zeichenkette: Die zu schreibende Zeichenkette.

GDOS_DIRECTION

legt das Format des GDOS-Ausdrucks für Rechenblätter und Graphen fest.

Syntax: GDOS_DIRECTION({Format})

Format:
- 0 - Hochformat
- 1 - Querformat
- -1 - keine Änderung (kann in einem Macro verwendet werden, wenn die vom Benutzer eingegebene Ausrichtung einen ungültigen Wert hat)

GDOS_FONT_SIZE

setzt den Zeichsatz und seine Größe, der Verwendung findet beim Drucken eines Blattes.

Syntax: GDOS_FONT_SIZE({"Zeichensatz"}, {Größe})

Zeichensatz: Name des Zeichensatzes

Größe: Größe der Zeichen in Punkten

z.B. GDOS_FONT_SIZE("SWISS", 20)

GDOS_GRAPH

druckt den Graphen im obersten Fenster mittels GDOS aus.

Syntax: GDOS_GRAPH({Gerät/"Gerät"}, {Exemplare})

Gerät: Die GDOS-Geräte-Nummer wird in der Datei "ASSIGN.SYS" benutzt. Wenn dieser Parameter fehlt, wird der Default-Wert 21 gesetzt, was einem normalen Drucker entspricht. Gerät kann auch eine Zeichenkette sein, wie "31:TEST.GEM". Es wird dann ein Metafile mit dem Namen "TEST.GEM" erzeugt. Die ersten beiden Buchstaben der Zeichkette geben die Gerätenummer an, die im Zahlenbereich 31..40 liegen muß. Auch diese Nummer muß in ASSIGN.SYS aufgeführt sein.

Exemplare: bestimmt, wieviele Kopien gedruckt werden. 1 ist Default.

z.B. GDOS_GRAPH(21, 1) startet einen Ausdruck auf dem normalen Drucker.

GDOS_IMAGE_SIZE

legt die Größe des GDOS-Ausdruckes auf dem Drucker in Druckereinheiten fest.

Syntax: GDOS_IMAGE_SIZE({x}, {y}, {b}, {h})

x, y, b, h: horizontale X-Position, vertikale Y-Position, Breite, Höhe des Ausdrucks auf dem Papier. Wenn ein Parameter mit -1 übergeben wird, so bleibt der Originalwert unverändert.

z.B. GDOS_IMAGE_SIZE(0, 0, 1000, 1000)

GDOS_IMAGE_SIZE(-1, 0, -1, 1000) läßt X-Position und Breite unverändert

GDOS_JUSTIFY

legt die horizontale und vertikale Ausrichtung bei GDOS-Ausdrucken fest.

Syntax: GDOS_JUSTIFY({horizontal}, {vertikal})

horizontal: waagerechte Ausrichtung des Ausdrucks:
0 - links, 1 - Mitte, 2 - rechts, -1 - unverändert

vertikal: senkrechte Ausrichtung:
0 - oben, 1 - Mitte, 2 - unten, -1 - unverändert

z.B. GDOS_JUSTIFY(1, 1) Der Ausdruck wird in der Mitte ausgegeben

GDOS_JUSTIFY(0, -1) richtet den Ausdruck nach links aus und läßt die vertikale Justierung unverändert.

GDOS_PAPER_SIZE

bestimmt die GDOS-Papiergröße des Druckers in Druckereinheiten.

Syntax: GDOS_PAPER_SIZE(x, y, b, h)

x, y, b, h: siehe GDOS_IMAGE_SIZE()

z.B. GDOS_PAPER_SIZE(-1, 10, 700, -1)

GDOS_SHEET

druckt ein Blatt mittels GDOS aus.

Syntax: GDOS_SHEET({Gerät},{Bereich},{Spaltenkopf}, {Zeilenkopf},{Gitter}, {Exemplare})

Gerät: die GDOS-Device-Nummer. (s. GDOS_GRAPH())

Bereich: Feldbereich, der ausgedruckt werden soll. Bei einem Fehlen dieses Parameters wird der markierte Bereich ausgedruckt.

Spaltenkopf/Zeilenkopf: bestimmt, ob die Spaltenköpfe (A, B, ..) bzw. die Zeilenköpfe (1, 2, ..) gedruckt werden: 0 - Nein, 1 - Ja

Gitter: sollen die Gitterlinien auch gedruckt werden? 0 - Nein, 1 - Ja

Exemplare: Anzahl der Ausdrucke, 1 ist Default.

z.B. GDOS_SHEET(21, A0:B15, 1, 1, 1, 1)

GDOS_SHEET("31:TEST.GEM", A0:B15, 1, 1, 1, 1) erzeugt ein Metafile

GIVE

lädt ein Rechenblatt eines bestimmten Formates von Diskette.

Syntax: GIVE({"Dateiname"}, {A0_Bereich/"RC_Bereich"}, {Format})

GLOBAL_RNG()

liefert den Bereich A0:IV8191, die maximale Ausdehnung eines Rechenblatts.

Syntax: GLOBAL_RNG()

GOSUB

ermöglicht es Ihnen, ein anderes Macro als Unterprogramm aus dem aktuellen Macro heraus aufzurufen.

Syntax: GOSUB("Name"/A0_Feld)

Name: Name des aufzurufenden Macros

A0_Feld: Feld, an dem das Unterprogramm starten soll

Nach dem Ende eines Unterprogramms (das geschieht, wenn es auf eine RETURN()-Anweisung stößt) wird die Ausführung des Macros an dem Feld unter dem GOSUB()-Befehl fortgesetzt.

	A	B	C
0	=SELECT(B5)	=SELECT(B6)	=HLINE(1)
1	=FORMULA(1)	=FORMULA(2)	=VLINE(1)
2	=GOSUB(B0)	=GOSUB(C0)	=RETURN()
3	=RETURN()	=RETURN()	

wird in folgender Reihenfolge durchlaufen:

A0,A1,A2,B0,B1,B2,C0,C1,C2,B3,A3.

Wenn der Inhalt von Feld C1 "=HALT()" wäre, würde die Ausführung des ganzen Macros bei C1 beendet werden.

GOTO

springt auf ein neues Feld im Macro-Blatt, an dem die Ausführung des Macros fortgesetzt wird.

Syntax: GOTO(A0_Feld)

z.B. GOTO(C5)

Mit der Funktion GOTO() können Sie Schleifen innerhalb Ihrer Macros erzeugen.

GRAPH_AXIS_FONT

Syntax: GRAPH_AXIS_FONT({"String1"}, {"String2"}, {Zahl1}, {"String3"}, {Zahl2})

GRAPH_COPY

Syntax: GRAPH_COPY({"alter_Name"}, "neuer_Name")

GRAPH_DATARNG_CREATE

Syntax: GRAPH_DATARNG_CREATE({"Name"},{A0_Bereich/ "RC_Bereich"},{Zahl1}, {Zahl2})

GRAPH_DATARNG_DELETE

Syntax: GRAPH_DATARNG_DELETE({"Name"}, {A0_Bereich/"RC_Bereich"/Zahl})

GRAPH_DATARNG_RESET

Syntax: GRAPH_DATARNG_RESET({"Name"})

GRAPH_DELETE

Syntax: GRAPH_DELETE({"Name"})

GRAPH_GRID

Syntax: GRAPH_GRID({"Name"}, {vertikal}, {horizontal})

GRAPH_KEY

Syntax: GRAPH_KEY({"Name"}, {Zahl1}, {"String"}, {Zahl2})

GRAPH_LABELRNG_DELETE

Syntax: GRAPH_LABELRNG_DELETE({"Name"})

GRAPH_LABELRNG_INSERT

Syntax: GRAPH_LABELRNG_INSERT({"Name"},
{A0_Bereich/"RC_Bereich"})

GRAPH_MOVE_KEY

Syntax: GRAPH_MOVE_KEY({"String"}, {Zahl1}, {Zahl2})

GRAPH_NAMERNG_DELETE

Syntax: GRAPH_NAMERNG_DELETE({"Name"},
A0_Bereich/"RC_Bereich")

GRAPH_NAMERNG_INSERT

Syntax: GRAPH_NAMERNG_INSERT({"Name"},
A0_Bereich1/"RC_Bereich1",
{A0_Bereich2/"RC_Bereich2"})

GRAPH_NEW

Syntax: GRAPH_NEW("Name", {Zahl}, {Zahl})

GRAPH_OPEN

öffnet das Fenster zu einem Graph.

Syntax: GRAPH_OPEN({"Name"}, "Graphname")

Name: Name des gewünschten Rechenblattes im Speicher.

Graphname: Name des Graphen

z.B. GRAPH_OPEN("TEST.SPD", "GRAPH1")

Das geöffnete Fenster kann man mit WIND_CLOSE() schließen.

GRAPH_PIE_LABELS

Syntax: GRAPH_PIE_LABELS({"Name",{zeige_Prozent}, {zeige_Wert})

GRAPH_RENAME

Syntax: GRAPH_RENAME({"alter_Name"}, "neuer_Name")

GRAPH_RESET()

Syntax: GRAPH_RESET()

GRAPH_SET_AXIS

Syntax: GRAPH_SET_AXIS({"Name"}, {Zahl1}, {Zahl2}, {Zahl3}, {Zahl4})

GRAPH_TEXTRNG_CREATE

Syntax: GRAPH_TEXTRNG_CREATE({"Name"}, {A0_Bereich/"RC_Bereich"})

GRAPH_TEXTRNG_DELETE

Syntax: GRAPH_TEXTRNG_DELETE({"Name"}, {A0_Bereich/"RC_Bereich"/Zahl})

GRAPH_TEXTRNG_RESET

Syntax: GRAPH_TEXTRNG_RESET({"Name"})

GRAPH_TYPE

Syntax: GRAPH_TYPE({"Name"}, {Zahl1}, {Zahl2})

HALT

bricht das aktuelle Macro und alle anderen Macros, die das aktuelle aufgerufen haben, ab. Die Kontrolle erhält der Benutzer, also Sie.

Syntax: HALT()

HEAD_FOOT

legt Kopf- und Fußzeilen fest und deren Abstand zum aktuellen Text.

Syntax: HEAD_FOOT({"Kopfzeile"}, {"Fußzeile"}, {Abstand_Kopf}, {Abstand_Fuß})

Kopfzeile/Fußzeile: Zeichenkette, welche die Zeichen enthält, die am Anfang bzw. Ende des Textes ausgegeben werden.

Abstand_Kopf/Abstand_Fuß: Anzahl der Leerzeilen zwischen aktuellem Text und Kopfzeile/Fußzeile.

z.B. HEAD_FOOT("Das ist die Kopfzeile", "Völlig am Ende", 1, 2)

HIDE_ZERO

bestimmt, ob Felder, welche nur die 0 enthalten, angezeigt werden sollen.

Syntax: HIDE_ZERO({Null_anzeigen})

Null_anzeigen: TRUE - ja, FALSE - nein.

HJUSTIFY (siehe Anmerkung zu den BLATT-Menü-Macros)

bestimmt die horizontale Ausrichtung innerhalb eines Feldes.

Syntax: HJUSTIFY({A0_Bereich/"RC_Bereich"/A0_Feld/"RC_Feld"}, {Typ}, {Ausrichtung})

Ausrichtung: in einem entsprechend großen Feld wird der Inhalt

0 - links
1 - in der Mitte
2 - rechts
3 - im Blocksatz (links- und rechtsbündig) (nur bei TEXT!) ausgegeben.

HLINE

blättert das Fenster horizontal eine angebbare Spaltenanzahl weiter.

Syntax: HLINE({Anzahl})

Anzahl: Anzahl Spalten, die weitergeblättert werden soll. Siehe VLINE().

HPAGE

blättert das Fenster horizontal eine angebbare Seitenanzahl weiter.

Syntax: HPAGE({Anzahl})

Anzahl: Anzahl Seiten, die weitergeblättert werden soll. Siehe VLINE().

INSERT_COLS

fügt in das Rechenblatt neue Spalten ein.

Syntax: INSERT_COLS("Spalte", Anzahl)

Spalte: Die Spaltenbezeichnung, ab der eingefügt werden soll, z.B. "A","IV".

Anzahl: Anzahl Spalten, die eingefügt werden sollen.

z.B. INSERT_COLS("B", 2) fügt an der Spalte B 2 neue Spalten ein.

Beachten Sie, daß Spalten nur eingefügt werden können, wenn am rechten Rand des Rechenblatts unbelegte Spalten vorhanden sind.

INSERT_ROWS

fügt neue Zeilen ins Rechenblatt ein.

Syntax: INSERT_ROWS(Zeile, Anzahl)

z.B. INSERT_ROWS(3, 2) fügt bei Zeile 3 zwei neue Zeilen ein.

Es gilt das bei INSERT_COLS() Beschriebene.

LEADING

legt die Prefixe fest.

Syntax: LEADING({"Prefix1"}, {"Prefix2"}, {"Prefix3"})

Prefix1/2/3: Diese drei Zeichenkette stellen die Prefixe dar, die Feldinhalten vorangestellt werden. Es werden jeweils nur die ersten drei Buchstaben verwendet. Fehlende Parameter belassen das zugehörige Prefix unverändert.

z.B. LEADING("$", , "") setzt das erste Prefix als "$", läßt das zweite unverändert und weist dem dritten einen Leerstring zu.

LINE_FEED

löst Zeilenvorschübe aus.

Syntax: LINE_FEED({Anzahl})

Anzahl: Anzahl der Zeilenvorschübe. (Default ist 1)

LOAD

Syntax: LOAD({"Dateiname"})

Dateiname: Name des Rechenblatts, welches geladen werden soll.

z.B. LOAD(): Die Dateiauswahlbox erscheint.

LOAD("TEST.SPD") oder LOAD("A:\KFZ\TEST.SPD")

LOAD_ASCII

lädt einen ASCII-Text von Diskette in das aktuelle Rechenblatt.

Syntax: LOAD_ASCII({"Dateiname"}, {A0_Bereich/"RC_Bereich"}, {Als_Zahl})

Dateiname: bestimmt die Datei, welche geladen werden soll.

A0_Bereich/RC_Bereich: Bereich des Rechenblatts, in den der Text geladen werden soll.

Als_Zahl: FALSE: Datei wird als Text gelesen. TRUE: Datei wird wie Zahlen behandelt.

z.B. LOAD_ASCII("TEST.DOC", A0:B15, 1)

LOAD_CFG

lädt eine neue Druckereinstellung.

Syntax: LOAD_CFG({Dateiname"})

Dateiname: Die Endung einer Konfigurationsdatei lautet normalerweise .CFG.

z.B. LOAD_CFG("EPSON.CFG")

LOAD_UDEF

liest neue Funktionen von Diskette in den Speicher.

Syntax: LOAD_UDEF({"Dateiname"}, {global}, {Auswahl})

Dateiname: gibt die zu ladende Datei an.

global: FALSE: Die neuen Funktionen sind nur dem aktuellen Blatt bekannt. (Default); TRUE: Die neuen Funktionen stehen allen Blättern zur Verfügung.

Auswahl: FALSE: Alle neuen Funktionen in der Datei werden geladen. (Default); TRUE: Bei jeder Funktion können Sie entscheiden, ob Sie geladen wird.

z.B. LOAD_UDEF("A:\TEST.UFN", ,)

LOCK_COL

ermöglicht es, die ganz linke Spalte des Fensters festzuhalten. Sie wird beim Blättern also nicht verschoben.

Syntax: LOCK_COL({festhalten})

festhalten: TRUE - ja, FALSE - wieder loslassen (Default)

LOCK_ROW

Die oberste Zeile im Fenster kann festgehalten werden.

Syntax: LOCK_ROW({festhalten})

festhalten: TRUE - ja, FALSE - wieder loslassen (Default)

MARGINS

setzt die Ränder links, rechts, oben und unten.

Syntax: MARGINS({links}, {rechts}, {oben}, {unten})

links: linker Rand. (0..rechts)

rechts: rechter Rand. (links..Papierbreite)

oben: oberer Rand. (0..unten)

unten: unterer Rand. (oben..Zeile pro Blatt)

z.B. MARGINS(5, 75, 5, 61) ergibt einen erlaubten Ausgabebereich, der jeweils 5 Zeichen von der Papierkante entfernt ist, wenn das Blatt 80*66 Zeichen groß ist.

MAT_INVERSE

invertiert eine Matrix.

Syntax: MAT_INVERSE(A0_Bereich1/"RC_Bereich1", A0_Bereich2/"RC_Bereich2"/A0_Feld/"RC_Feld")

A0_Bereich1/"RC_Bereich1": die zu invertierende Matrix

A0_Bereich2/"RC_Bereich2"/A0_Feld/"RC_Feld": Ausgabebereich

z.B. MAT_INVERSE(A0:B1, C6:D7)

MAT_MULTIPLY

multipliziert zwei Matrizen miteinander.

Syntax: MAT_MULTIPLY(A0_Bereich1/"RC_Bereich1", A0_Bereich2/"RC_Bereich2", A0_Bereich3/"RC_Bereich3"/A0_Feld/"RC_Feld")

A0_Bereich1/"RC_Bereich1": erste Matrix

A0_Bereich2/"RC_Bereich2": zweite Matrix

A0_Bereich3/"RC_Bereich3"/A0_Feld/"RC_Feld": Ausgabebereich

z.B. MAT_MULTIPLY(A0: B1, D0: E1, C6)

MOVE

verschiebt ein Feld oder einen Bereich auf dem Blatt. Es funktioniert exakt wie COPY(), nur daß nach dem Kopieren der Quellbereich gelöscht wird.

Syntax: MOVE({A0_Bereich1/"RC_Bereich1"/A0_Feld1/"RC_Feld1"}, {A0_Bereich2/"RC_Bereich2"/A0_Feld2/"RC_Feld2"}, {ausfüllen}, {Ergebnisse})

OCCUPIED_RNG()

liefert den kleinsten Bereich, der alle belegten Felder eines Rechenblatts enthält.

Syntax: OCCUPIED_RNG()

z.B. Wenn alle nichtleeren Felder zwischen Spalte D und T und zwischen Zeile 4 und 12 liegen, so liefert diese Funktion A0:T12. Es wird also der Bereich zurückgegeben, dessen linkes oberes Feld stets A0 ist, das rechte untere Feld ist das Feld, welches im Rechenblatt am weitesten rechts und unten liegt.

PAPER_SIZE

legt die Papiergröße.

Syntax: PAPER_SIZE({Breite}, {Höhe})

Breite: Blattbreite in Zeichen, normal 80.

Höhe: Blatthöhe in Zeiche, normal 66.

z.B. PAPER_SIZE(80, 66)

PARSE

löst das Aufspalten eines Bereichs aus.

Syntax: PARSE ({A0_Bereich1/"RC_Bereich1"}, A0_Bereich2/"RC_Bereich2" / A0_Feld/"RC_Feld")

A0_Bereich1/"RC_Bereich1": Bereich, der aufgespalten werden soll, einschließlich des Feldes, das den Format-String enthält.

A0_Bereich2/"RC_Bereich2"/A0_Feld/"RC_Feld": Ausgabebereich

z.B. PARSE(A0:A12, C5) trennt die Felder A0..A12 auf, und gibt dies in die Felder unterhalb C5 aus.

PARSE_GEN

analysiert den Bereich, welcher aufgespalten werden soll.

Syntax: PARSE_GEN({A0_Feld/"RC_Feld"})

A0_Feld/"RC_Feld": linkes oberes Feld des aufzuspaltenden Bereichs. Es wird über diesem Feld eine Zeile mit dem Format-String eingefügt.

z.B. wenn A0 "12 13 14" enthält, liefert PARSE_GEN(A0) in A0 "Z Z Z", in A1 steht "12 13 14".

PASSWORD

läßt Sie ein neues Password für eine Datei eingeben oder ändern.

Syntax: PASSWORD({"altes_Wort"}, {"neues_Wort"})

altes_Wort: Das aktuelle Password. Kann ausgelassen werden, falls keines existiert.

neues_Wort: Das neue Password. Kann ausgelassen werden, wenn Sie den Password-Schutz entfernen möchten. Dazu müssen Sie aber das Password in "altes_Wort" übergeben haben.

PRIME

belegt Zahlen- und Formel-Felder in einem angebbaren Bereich mit einem Wert.

Syntax: PRIME({A0_Bereich/"RC_Bereich"}, {Wert})

Wert: Zahl, die in die Felder geschrieben wird. Wenn diese Angabe fehlt, wird 0 eingesetzt.

z.B. PRIME(A0:B15, 12) weist allen Feldern im Bereich A0..B15, die schon Zahlen oder Formeln enthalten, den Zahlenwert 12 zu.

PRINT_CONTENTS

druckt die Feldinhalte eines angebbaren Feldbereichs.

Syntax: PRINT_CONTENTS("Device", {A0_Bereich/"RC_Bereich"}, {Datentyp})

Device: "PRN:" - Parallelport
"AUX:" - Serieller Port

Datentyp: nur die entsprechenden Felder werden ausgedruckt, Kombinationen sind durch einfache Addition möglich.

1 - Zahlen
2 - Formeln
3 - Texte
7 - druckt ALLE 3 Typen:Zahlen, Formeln und Texte

PRINT_DISK

leitet die Druckausgabe in eine Datei auf Diskette um.

Syntax: PRINT_DISK({"Dateiname"}, {A0_Bereich/"RC_Bereich"})

PRINT_DIVIDERS

legt fest, welche Zeichen als horizontale und vertikale Trennzeichen verwendet werden.

Syntax: PRINT_DIVIDERS({h_Code}, {h_an}, {v_Code}, {v_an})

h_Code/v_Code: ASCII-Code (32..255) des horizontalen/vertikalen Trennzeichens

h_an/v_an: soll das horizontale/vertikale Trennzeichen ausgegeben werden? TRUE - ja, FALSE - nein.

Wenn ein Parameter weggelassen wird, bleiben die ursprünglichen Einstellungen erhalten.

z.B. PRINT_DIVIDERS(CODE("+"), 1, ,) setzt das "+" als horizontales Trennzeichen.

PRINTER_DETAILS

bestimmt einige Eigenschaften des Druckers.

Syntax: PRINTER_DETAILS ({ Port } , { Breite } , { Papierart } , {Zeilenende})

Port: Anschlußport des Druckers:
- 0 - Parallelport
- 1 - Serieller Port

Breite: Druckerbreite (benutzt bei PRINT_SIDEWAYS()):
- 0 - 10 Zoll
- 1 - 15 Zoll

Papierart:
- 0 - Endlospapier
- 1 - Einzelblatt

Zeilenende: Zeichenfolge am Ende einer Zeile:
- 0 - CR (Carriage Return)
- 1 - LF (Line Feed)
- 2 - CR/LF

z.B. PRINTER_DETAILS(0, 0, 0, 2) ist die normale Einstellung.

PRINT_OUTPUT

bestimmt die Art der Ausgabe. Die einzelnen Parameter können dabei nur folgende Werte und Bedeutungen beinhalten: TRUE - ja, FALSE - nein.

Syntax: PRINT_OUTPUT({Formfeed}, {Seitenpause}, {Spaltenkopf}, {Zeilenkopf})

Formfeed:	Soll nach Ende der Druckausgabe ein Seitenvorschub ausgelöst werden?
Seitenpause:	Soll am Seitenende eine Pause eingeleget werden?
Spaltenkopf:	Spaltenköpfe mitausdrucken?
Zeilenköpfe:	Zeilenköpfe mitausdrucken?

PRINT_SHEET

druckt einen angebaren Feldbereich aus.

Syntax: PRINT_SHEET("Device", {A0_Bereich/"RC_Bereich"})

Device: Ausgabegerät am:
"PRN:" - Parallelen Port
"AUX:" - Seriellen Port

PRINT_SIDEWAYS

druckt einen angebbaren Feldbereich um 90 Grad gedreht aus.

Syntax: PRINT_SIDEWAYS("Device", {A0_Bereich/"RC_Bereich"}, Zeilenanzahl)

Device: "PRN:" - Parallelport
"AUX:" - Serieller Port

Zeilenanzahl: Ein Index gibt an, wieviele Zeilen des Bereichs (also wieviel Zeichen pro Druckerzeile) gedruckt werden:
0 - 40 Zeilen
1 - 55
2 - 60
0..2 besonders für DIN A4 (10-Zoll)-breite Drucker.
3 - 66
4 - 82
5 - 99
3..5 für DIN A3 (15-Zoll)-breite Drucker.

z.B. PRINT_SIDEWAYS("PRN:", "R0C0:R5C5", 1)

QUIT()

verläßt das Programm K-Spread 4.

Syntax: QUIT()

Es findet eine Sicherheitsabfrage statt, ob Sie K-Spread wirklich verlassen wollen.

RECALC_RANGE

legt den Bereich fest, der neuberechnet werden soll.

Syntax: RECALC_RANGE({A0_Bereich/"RC_Bereich"})

Wenn kein Parameter übergeben wird, setzt das Programm A0:IV8191.

REFTEXT

wandelt eine Feldbezeichnung (A0_Koordinaten) in eine Zeichenkette, welche die absolute Bezeichnung repräsentiert.

Syntax: REFTEXT(A0_Feld, Typ)

Typ: FALSE (=0): Funktion liefert RC_Koordinaten zurück
TRUE (<>0): Funktion liefert A0_Koordinaten zurück

z.B. REFTEXT(A0, 0) ergibt "R0C0"
REFTEXT(A0, 1) ergibt "A0"

Diese Funktion ist sehr nützlich, wenn Sie die augenblickliche Position des Cursors testen möchten:

z.B. =IF(REFTEXT(ACTIVE_CELL(),1) = "D15", THEN RETURN(),
ELSE SELECT("R[1]C[0]")
=...
=...
=GOTO("R[-3]C")

Bemerkung: ACTIVE_CELL() liefert die A0_Koordinaten des Feldes zurück, auf dem der Edit-Rahmen steht.

RELREF

liefert die Beziehung in RC_Koordinaten von zwei Feldern, die in A0_Koordinaten angegeben werden.

Syntax: RELREF(A0_Feld1, A0_Feld2)

z.B. RELREF(A0, B1) ergibt "R[-1]C[-1]"
RELREF(E6, D4) ergibt "R[2]C[1]"

RENAME_DSET

benennt ein existierendes Datenblatt um.

Syntax: RENAME_DSET("alter_Name", "neuer_Name")

alter_Name: Zu ändernder Name des Blattes.

Beachten Sie, daß die Endung weder .SPD noch .SPM muß.

z.B. RENAME_DSET("Test", "Rechnung")

RESET_RNAME()

löscht ALLE Bereichsnamen des aktuellen Rechenblattes.

Syntax: RESET_RNAME()

RESET_TRACE()

löscht ALLE Überwachungen im aktuellen Rechenblatt.

Syntax: RESET_TRACE()

RESET_UDEF

entfernt ALLE Neuen Funktionen der angebbaren Liste aus dem Speicher.

Syntax: RESET_UDEF({global})

global: es werden ALLE Neuen Funktionen gelöscht, die TRUE: in allen Blättern verwendet werden können; FALSE: nur dem aktuellen Rechenblatt bekannt sind (Default).

RETURN

kennzeichnet das Ende eines Macros. Die Kontrolle wird an den Aufrufer des Macros zurückgegeben. Das können etwa Sie sein, oder ein anderes Macro.

Syntax: RETURN()

ROW_HEIGHT

setzt eine neue Zeilenhöhe.

Syntax: ROW_HEIGHT({A0_Bereich/"RC_Bereich"}, {Höhe})

Höhe: 0 - versteckt die Zeilen
1..16 - die neue Höhe

Ansonsten gilt das bei COL_WIDTH() Erklärte.

RUN

erlaubt Ihnen, aus einem Macro ein anderes Macro aufzurufen. Es bewirkt das gleiche wie das Anwählen von "Macro ausführen.." im MACRO-Menü.

Syntax: RUN(A0_Feld/"RC_Feld")

A0_Feld/"RC_Feld": Der Name eines Macros oder das Feld, an dem begonnen werden soll. Der Name des Macro-Blattes steht vor dieser Angabe, durch "!" getrennt.

z.B. RUN(TEST!B20) und
RUN("TEST!R[20]C[1]") bewirken dasselbe.

SAFETY

legt fest, ob Warnmeldungen erscheinen sollen beim Überschreiben oder Löschen von Daten.

Syntax: SAFETY({Überschreiben}, {Löschen})

Überschreiben: TRUE - Warnmeldung erscheint beim Überschreiben von Feldinhalten; FALSE - keine Meldung

Löschen: TRUE - Warnmeldung beim Löschen von Feldinhalten; FALSE - keine Meldung

z.B. SAFETY(1, 1) Warnungen erscheinen bei beiden Vorgängen
SAFETY(1,) Meldungen gibt es beim Überschreiben, die Einstellung für das Löschen bleibt unverändert.

SAVE

Syntax: SAVE("Dateiname", "Name")

Name: Name des Rechenblatts, welches gespeichert werden soll.

Dateiname: Pfad mit Dateiname, über welchen das Rechenblatt gespeichert werden soll.

z.B.: SAVE("TEST.SPD",
speichert das Rechenblatt "TEST.SPD" unter diesem Dateinamen in das aktuelle Unterverzeichnis.

z.B. SAVE("A:\KFZ\ABRECHNUNG.SPD", "TEST.SPD")
sichert das Blatt "TEST.SPD" in den Ordner "KFZ" als "ABRECHNUNG.SPD".

Wenn Sie einen Dateinamen angeben, welcher nicht mit dem Namen des Rechenblatts übereinstimmt, so wird der Dateiname der neue Name des Rechenblatts. Wollen Sie dies vermeiden, geben Sie folgendes ein:
=SAVE_RANGE({"Dateiname"}, {"Name"}, OCCUPIED_RNG(),
{Speichere_Formeln})

SAVE_DESKTOP()

sichert die Arbeitsumgebung auf Diskette.

Syntax: SAVE_DESKTOP()

SAVE_RANGE

Syntax: SAVE_RANGE({"Dateiname"}, {"Name"}, {"RC_Bereich"}, {Speichere_Formeln})

Name: Name des Rechenblatts, aus welchem ein Bereich gespeichert werden soll.

Dateiname: Pfad und Dateiname, über welche der Bereich gespeichert werden soll.

RC_Bereich: Der zu speichernde Feldbereich. Wenn Sie nichts angeben, werden die momentan selektierten Felder gespeichert.

Speichere_Formeln: FALSE (=0): Formeln werden nicht gespeichert, sondern nur deren Ergebnisse. (Default) TRUE (<>0): Formeln werden gespeichert. Zahlen und Text bleiben von "Speichere_Formeln "unberührt.

SAVE_UDEF

speichert Neue Funktionen auf Diskette.

Syntax: SAVE_UDEF({"Dateiname"}, {global}, {Auswahl})

Die Parameter haben die gleiche Bedeutung wie bei LOAD_UDEF.

z.B. SAVE_UDEF("A: \TEST.UFN", 1,)

SELECT

setzt den Edit-Rahmen und/oder den Feld-Marker auf eine neue Position.

Syntax: SELECT({A0_Bereich/"RC_Ber."}{,}{A0_Feld/"RC_Feld"})

A0_Bereich/"RC_Ber.": neuer Bereich für die selektierten Felder (=Feld-Marker)

A0_Feld/"RC_Feld": neue Position des Edit-Rahmens

z.B. SELECT(A0:B5, C6) setzt beide Cursor neu
SELECT(A0:B5) setzt nur den Feld-Marker neu
SELECT(C6) setzt nur den Edit-Rahmen neu

Wie Sie sehen, brauchen Sie keine trennende Kommas, wenn nur ein Cursor gesetzt werden soll.

SELECT_LAST()

setzt die Cursor auf die vorhergehende Position.

Syntax: SELECT_LAST()

SET_TOF()

bestimmt den Anfang der Ausgabe ("Top Of Form").

Syntax: SET_TOF()

SET_VALUE

setzt den Inhalt eines Feldes in einem Macro-Blatt neu.

Syntax: SET_VALUE(A0_Feld/"RC_Feld", Zahl/Text)

Zahl/Text: Eine Zahl (kann auch das Ergebnis einer Formel sein) oder ein Text, den das angesprochene Feld erhalten soll.

z.B. SET_VALUE(A6, 100)
SET_VALUE(A6, A6+1) erhöht die Zahl in Feld A6 um 1
SET_VALUE(A6, "Text")

SHOW

legt fest, ob Formeln oder ihr Ergebnis angezeigt werden, bzw. ob die Datentypen der einzelnen Felder angezeigt werden.

Syntax: SHOW({Formeln_anzeigen}, {Datentyp_anzeigen})

Formeln_anzeigen: TRUE - ja, FALSE - nein

Datentyp_anzeigen: TRUE - ja, FALSE - nein

Wenn ein Parameter weggelassen wird, bleibt die zugehörige Anzeigeart unverändert. Beachten Sie bitte, daß es nicht sinnvoll ist, für beide Parameter TRUE zu setzen. Ist dies trotzdem der Fall, so hat "Zeige Formeln" Vorrang.

z.B. SHOW(1,0) zeigt Formeln
SHOW(0,1) zeigt den Datentyp
SHOW(1,1) zeigt Formeln

SORT

sortiert einen Bereich.

Syntax: SORT({A0_Ber./"RC_Bereich"}, Zeile1/"Spalte1", {Zahl2}, {Zeile2/"Spalte2"}, {Zahl4}, Zahl5)

STEP

schaltet in den Einzelschrittmodus für Macros. In dieser Betriebsart erscheint bei jedem zu bearbeitenden Feld eine Meldungsbox am Bildschirm, die angibt, welches Feld gerade bearbeitet wird. Diese Box

können Sie verlassen, indem Sie angeben, ob Sie in diesem Modus oder ohne ihn fortfahren möchten, oder ob Sie die ganze Macro-Ausführung beenden wollen.

Syntax: STEP()

z.B.

	A
0	=STEP()
1	=SELECT(D5)
2	=RETURN()

TABLE_ONE

erlaubt die Verwendung eines Eingabebereichs 1.

Syntax: TABLE_ONE({A0_Bereich/"RC_Bereich"}, {Ausgangsfeld})

A0_Bereich/"RC_Bereich": Bereich, in dem sich die Liste mit den Ausgangswerten befindet.

Ausgangsfeld: In dieses Feld werden nacheinander die verschiedenen Werte eingetragen, und jeweils anschließend eine Neuberechnung ausgelöst. Wenn diese Angabe fehlt wird das Feld eingesetzt, auf dem der Edit-Rahmen steht.

z.B. TABLE_ONE(A0: C15, E7)

TABLE_TWO

erlaubt die Verwendung eines Eingabebereichs 2.

Syntax: TABLE_TWO({A0_Bereich/"RC_Bereich"}, Ausgangsfeld1, Ausgangsfeld2)

Diese Funktion ist TABLE_ONE() sehr ähnlich. Der wesentliche Unterschied besteht darin, daß jetzt zwei Felder angegeben werden können, die die Ausgangswerte erhalten.

TAKE

speichert ein Rechenblatt in einem angebbaren Format auf Diskette.

Syntax: TAKE({"Dateiname"}, {"RC_Bereich"}, {Format})

TEXT_COLOUR (siehe Anmerkung zu den BLATT-Menü-Macros)

legt die Textfarbe fest. Sie können -wie soeben erwähnt- entweder die aktuellen Felder ändern, oder eine Grundeinstellung vornehmen.

Syntax: TEXT_COLOUR({A0_Bereich/"RC_Bereich"/A0_Feld/" RC_Feld"}, {Typ}, {Farbe})

A0_Bereich/"RC_Bereich"/A0_Feld/"RC_Feld": Zu ändernde Felder. Wenn Sie nur den Default festlegen wollen (s. Typ), wird dieser Parameter ignoriert.

Typ: Datentyp der Felder, die geändert werden sollen.

1 - Zahlen
2 - Formeln
4 - Text
8 - setzt den Default der obigen Datentypen (1, 2, bzw. 4).

Um beliebige Kombinationen zu erreichen, addieren Sie einfach die gewünschten Typenwerte, z.B.

1 - nur bei Feldern mit Zahlen
2 - Formeln
4 - Texten
3 - Zahlen und Formeln
6 - Formeln und Text
9 - Grundeinstellung nur für Felder mit Zahlen
10 - Formeln
11 - Zahlen und Formeln

Wenn der Wert 8 (Default setzen) verwendet wird, sind von den Festlegungen die derzeit belegten Felder NICHT betroffen. Erst auf neu einzugebende Felder wirken sich die Einstellungen aus.

Farbe: neue Textfarbe:

0 - Weiß
1 - Schwarz
2 - Rot (Farbe kann mit dem Kontrollfeld (Accessory) geändert werden)
3 - Grün (Farbe kann mit dem Kontrollfeld (Accessory) geändert werden)

z.B. TEXT_COLOUR(A0:A4, 1, 1) bewirkt, daß die Zahlen in den Feldern A0..A4 schwarz ausgegeben werden.

TEXTREF

wandelt eine Feldbezeichnung, die in einer Zeichenkette vorliegt, in einen Standardverweis um.

Syntax: TEXTREF("Koordinaten", Typ)

"Koordinaten": Zeichenkette, die A0_Koordinaten oder RC_Koordinaten enthält

Typ: FALSE: Zeichenkette enthält RC_Koordinaten
TRUE: A0_Koordinaten

z.B. TEXTREF("A0", 1) ergibt A0
TEXTREF("R0C0", 0) A0
TEXTREF("A0", 0) #REF; Fehler, denn A0 ist keine RC_Koordinate

TIDY_DESKTOP()

räumt die Arbeitsumgebung wieder auf.

Syntax: TIDY_DESKTOP()

TOP_NAME()

Syntax: TOP_NAME()

TRAILING

setzt die Suffixe.

Syntax: TRAILING({"Suffix1"}, {"Suffix2"}, {"Suffix3"})

Es gilt entsprechend das bei LEADING() Erwähnte.

TRANSPOSE

"kippt" einen Bereich, d.h. verwandelt die horizontale Anordnung in die vertikale, und umgekehrt.

Syntax: TRANSPOSE({A0_Bereich1/"RC_Bereich1"},
{A0_Bereich2/"RC_Bereich2"/A0_Feld/"RC_Feld"})

A0_Bereich1/"RC_Bereich1": Bereich, der gekippt werden soll

A0_Bereich2/"RC_Bereich2"/A0_Feld/"RC_Feld": Bereich, in den der gekippte Ausgangsbereich ausgegeben wird. Bei Angabe eines Feldes benutzt das Programm alle benötigten Felder, die rechts und unterhalb dieses Feldes liegen.

Wenn dieser Parameter fehlt, werden die markierten Felder zu Ausgabe benutzt.

z.B. TRANSPOSE(A0:C5, E6)
TRANSPOSE(A0:C5,)

VIEW_GRID

erlaubt es Ihnen, im aktuellen Fenster das Gitter ein- und auszuschalten.

Syntax: VIEW_GRID({view}, {Gitter_anzeigen})

Gitter_anzeigen: TRUE - ja, FALSE - nein.

VJUSTIFY (siehe Anmerkung zu den BLATT-Menü-Macros)

legt die vertikale Ausrichtung in einem Feld fest.

Syntax: VJUSTIFY({A0_Bereich/"RC_Bereich"/A0_Feld/"RC_Feld"}, {Typ}, {Ausrichtung})

Ausrichtung: in einem entsprechend großen Feld wird der Inhalt

0 - oben

1 - in der Mitte

2 - unten ausgegeben.

VLINE

blättert das Fenster vertikal eine angebbare Zeilenanzahl weiter.

Syntax: VLINE({Anzahl})

Anzahl: Anzahl der Zeilen, die weitergeblättert werden soll. Wenn Anzahl größer als 0 ist, wird nach unten, ist Anzahl negativ, wird nach oben geblättert. Wenn Sie nichts übergeben, setzt das Programm als Default-Wert 1 ein.

z.B. VLINE(2) blättert zwei Zeilen nach unten

VLINE(-2) nach oben

VLINE() eine Zeile nach unten

VPAGE

blättert das Fenster vertikal eine angebbare Seitenanzahl weiter.

Syntax: VPAGE({Anzahl})

Anzahl: Anzahl Seiten, die weitergeblättert werden soll. Es gilt auch hier das unter VLINE() beschriebene.

WIND_CLOSE()

schließt das oberste dargestellte Fenster.

Syntax: WIND_CLOSE()

WIND_FULL

zeichnet ein Fenster in voller oder wieder in seiner urprünglichen Größe. Diese Funktion hat dieselbe Wirkung wie das Anklicken des Vollfeldes in der rechten oberen Ecke des Fensters.

Syntax: WIND_FULL(voll)

voll: FALSE (=0): Fenster schrumpft auf die Größe, die es vor der Volldarstellung hatte.

TRUE (<>0): Fenster wird in voller Größe dargestellt.

WIND_MOVE

zeichnet das oberste dargestellte Fenster an einer neuen Position.

Syntax: WIND_MOVE(x, y)

x, y: Koordinaten des linken oberen Ecks des Fensters

z.B. WIND_MOVE(100,50) zeichnet das Fenster an der Stelle neu, die 100 Bildschirmpunkte rechts und 50 unterhalb der linken oberen Ecke der Arbeitsfläche liegt.

WIND_OPEN

öffnet ein neues Fenster eines bestehenden Rechen- oder Macro-Blattes.

Syntax: WIND_OPEN("Name")

Name: Name des zu öffnenden Blattes.

z.B. WIND_OPEN("RECHNUNG") stellt ein neues Fenster zu RECHNUNG.SPD dar.

WIND_OPEN("MACRO.SPM") stellt ein Macro-Blatt dar.

WINDOW_RNG()

liefert die Ausdehnung des aktuellen Fensters.

Syntax: WINDOW_RNG()

WIND_RANGE

legt die Ausdehnung eines Fensters in Feldern fest.

Syntax: WIND_RANGE([A0_Bereich])

A0_Bereich: gibt an, wieviele Felder breit und hoch das aktuelle Fenster sein soll. Die kleinste Ausdehnung beträgt 50 auf 50 Felder. Dieser Wert wird auch gesetzt, falls Sie den Parameter weglassen.

z.B. WIND_RANGE(A0:AZ60) ist 52 Felder breit und 61 Felder hoch.

WIND_SIZE

zeichnet das oberste dargestellte Fenster mit neuer Größe.

Syntax: WIND_SIZE(Breite, Höhe)

Breite/Höhe: horizontale und vertikale neue Ausdehnung in Bildschirmpunkten

z.B. WIND_SIZE(250, 100) Das Fenster erhält die neue Ausdehnung 250 * 100.

WIND_TOP

aktiviert ein Rechenblatt, d.h. bringt dessen Fenster auf dem Bildschirm ganz nach oben.

Syntax: WIND_TOP("Name")

Name: Name des Rechenblatts, welches ganz oben dargestellt werden soll. Gibt es zu einem Rechenblatt mehrere Fenster, so kann hinter dem Name noch die gewünschte Nummer angegeben werden.

z.B. WIND_TOP("RECHNUNG:2") aktiviert das dritte Fenster (:0, 1, 2) von RECHNUNG.SPD

Diese Funktion erlaubt es Ihnen, aus einem Macro heraus ein beliebiges Rechenblatt-Fenster anzusprechen. Dies ist besonders wichtig, da nur auf Felder des aktuellen Rechenblatts zugegriffen werden kann.

WITH_INFILE (Siehe "Anm. zu Macros für erw. Dateihandhabung") öffnet eine Datei, um Sie als ASCII-Datei zu lesen.

Syntax: WITH_INFILE("Dateiname", Feld)

Dateiname: Pfad/Name der zu lesenden Datei

Feld: gibt das Feld an, ab dem das Macro, welches die Datei einliest, beginnt.

WITH_INOUTFILE (Siehe "Anm. zu Macros für erw. Dateihandhabung") öffnet eine Datei, um als ASCII-Datei aus ihr zu lesen oder in sie zu schreiben.

Syntax: WITH_INOUTFILE("Dateiname", A0_Feld/"RC_Feld")

Dateiname: Pfad/Name der Datei, aus der gelesen oder in die geschrieben wird

Feld: Startfeld des Macros, welches das Lesen oder Schreiben steuert.

WITH_OUTFILE (Siehe "Anm. zu Macros für erw. Dateihandhabung") öffnet eine Datei, um in diese als ASCII-Datei zu schreiben.

Syntax: WITH_OUTFILE("Dateiname", Feld)

Dateiname: Pfad/Name der zu schreibenden Datei

Feld: Startfeld des Macros, welches den Schreibvorgang vornimmt.